REMARKS

Reconsideration of the above-identified patent application in view of the amendments above and the remarks following is respectfully requested.

Claims 1-22 are in this case. Claims 15 and 20-22 have been rejected under § 101. Claims 1-22 have been rejected under § 103(a). Independent claim 15 has been canceled. Independent claims 1, 11, 16, 17 and 20 have been amended.

Specifically, claims 1, 11, 16, 17 and 20 have been amended to recite the limitation that the monitoring is for (a) suspicious portion(s) of data in a portion of the stream of data traffic that is expected to lack executable code. Support for these amendments is found in the specification at least on page 8 line 30 through page 9 line 5:

> Suspicious data includes data inconsistent with the protocol in use. For instance, an illegal character is filtered out of a stream of data traffic, *e.g.* HTTP traffic. The illegal character is unexpected as part of the protocol definition in the input data stream. Another option of such filtering is to treat all parts of the protocol as suspicious except for parts of the protocol that were specifically marked by the protocol to contain executable code. For example, an URL of a HTTP request is suspicious since any executable code segment found within it will constitute a worm. (emphasis added)

Note that page 6 lines 24-29 defines the stream of data traffic as possibly including both "legitimate executable code" (page 6 line 29) and "non-executable code, i.e. HTML" (page 6 line 26).

§ 101 Rejections

The Examiner has rejected claims 15 and 20-22 under § 101, as directed to non-statutory subject matter. The Examiner's rejection is respectfully traversed.

Claim 15 has been canceled, thereby rendering moot the Examiner's rejection of this claim.

With regard to the rejection of claims 20-22, contrary to the Examiner's interpretation of these claims, the scope of these claims includes both software embodiments and hardware embodiments. It is well known in the art that hardware and software are equivalent. See *e.g.* Andrew S. Tanenbaum, *Structured Computer Organization, 4th Edition* (Prentice-Hall International, 1998) page 8 (copy attached):

> *Hardware and software are logically equivalent.*
> Any operation performed by software can also be built directly into the hardware, preferably after it is sufficiently well understood. As Karen Panetta Lentz put it: "Hardware is just petrified software." Of course, the reverse is also true" any instruction executed by the hardware can also be simulated in software. The decision to put certain functions in hardware and others in software is based on such factors as cost, speed, reliability and frequency of expected changes. There are few hard and fast rules to the effect that X must go into the hardware and Y must be programmed explicitly. These decisions change with trends in technology and computer usage. (emphasis in original)

The specification as filed states explicitly, on page 6 lines 7-9:

> ...it is to be understood that the invention is not limited in its application to the details of construction and the arrangement of the components set forth in the following description or illustrated in the drawings.

on page 6 lines 15-17:

> It is important, therefore, that the claims be regarded as including such equivalent constructions insofar as they do not depart from the spirit and scope of the present invention.

and on page 12 lines 1-2:

> ...and accordingly, all suitable modifications and equivalents may be resorted to, falling within the scope of the invention.

It follows that the scope of claims 20-22 includes not just the software embodiments described in the specification but also their hardware equivalents.

§ 103(a) Rejections – Vella '913 in view of Schmall

The Examiner has rejected claims 1, 3 and 20 under § 103(a) as being unpatentable over Vella, US Patent Application Publication No. 2003/0212913 (henceforth, "Vella '913") in view of Schmall, "Classification and identification of malicious code based on heuristic techniques using Meta languages" (Hamburg 2003) (henceforth, "Schmall"). The Examiner's rejection is respectfully traversed.

Schmall is cited by the Examiner only as teaching the limitations of steps (c) and (d) of claim 1 and the functionality of element (c) of claim 20. Therefore, the following argument concentrates on Vella '913.

Vella '913 teaches a system **10** for detecting malicious code in executable attachments of e-mail and in downloaded executable files. An electronic mail analyzer **11** identifies executable attachments of e-mail and forwards the executable attachments to an executable file analyzer **13** for analysis. A download analyzer **14** identifies executable files in downloads and forwards the executable files to executable file analyzer **13** for analysis.

In other words, Vella '913 monitors a stream of data traffic specifically for e-mail attachments and downloaded files that are known a priori to be executable even before the attachments and files are inspected by executable file analyzer **13**. By contrast, the present invention, as recited in independent claims 1 and 20 as now amended, monitors a stream of data traffic for suspicious data in a portion of the stream of data traffic that is specifically expected *not* to include executable code.

In order for independent claims 1 and 20 to be unpatentable over Vella '913 in view of Schmall, these references must teach or suggest every recited limitation. As the Board of Patent Appeal and Interferences has confirmed in *In re Wada and Murphy*, Appeal 2007-3733,

> When determining whether a claim is obvious, an examiner must make "a searching comparison of the claimed invention – *including all its limitations* – within the teaching of the prior art". *In re Orchiai*, 71 F.3d 1565, 1572 (Fed. Cir. 1995) (emphasis added). Thus, "Obviousness requires a suggestion of all limitations in a claim." *CFMT, Inc. v. Yieldup Intern. Corp.*, 349 F.3d 1333, 1342 (Fed. Cir. 2003) (citing *In re Royka*, 490 F.2d 981, 985 (CCPA 1974)).

In the present case, neither Vella '913 nor Schmall teach, hint or suggest monitoring a stream of data traffic for suspicious data in a portion of the stream of data traffic that is expected, a priori, not to include executable code. It follows that independent claims 1 and 20 are allowable in their present form over the prior art cited by the Examiner.

With independent claim 1 allowable in its present form it follows that claim 3 that depends therefrom also is allowable.

§ 103(a) Rejections – Vella '913 in view of Schmall and further in view of Muttik '780

The Examiner has rejected claims 11, 16 and 17 under § 103(a) as being unpatentable over Vella '913 in view of Schmall and further in view of Muttik, US Patent No. 6,775,780 (henceforth, "Muttik '780"). The Examiner's rejection is respectfully traversed.

Muttik '780 teaches an emulator **110** for detecting malicious code in code **108** that has been introduced to a computer system **106**. As in the case of Vella '913, code **108** is known a priori to be executable. Therefore, the arguments above that demonstrate the allowability of independent claims 1 and 20 over the prior art cited by the Examiner also show, *mutatis mutandis*, that independent claims 11, 16 and 17 also are allowable in their present form over the prior art cited by the Examiner.

§ 103(a) Rejections – Vella '913 in view of Schmall and Muttik '780 and further in view of Shipley '236

The Examiner has rejected claims 2, 6, 7, 14 and 15 under § 103(a) as being unpatentable over Vella '913 in view of Schmall and Muttik '780 and further in view of Shipley, US Patent No. 6,119,236. The Examiner's rejection is respectfully traversed.

Claim 15 has been canceled, thereby rendering moot the Examiner's rejection of this claim.

It has been demonstrated above that claims 1 and 11 are allowable in their present form. It follows that claims 2, 6, 7 and 14 that depend therefrom also are allowable.

§ 103(a) Rejections – Vella '913 in view of Schmall and further in view of Touboul '194

The Examiner has rejected claim 4 under § 103(a) as being unpatentable over Vella '913 in view of Schmall and further in view of Touboul, US Patent No. 6,092,194. The Examiner's rejection is respectfully traversed.

It has been demonstrated above that claim 1 is allowable in its present form. It follows that claim 4 that depends therefrom also is allowable.

§ 103(a) Rejections – Vella '913 in view of Schmall and Muttik '780 and further in view of Made '076

The Examiner has rejected claims 5, 8, 9, 12, 13, 18, 19, 21 and 22 under § 103(a) as being unpatentable over Vella '913 in view of Schmall and Muttik '780 and further in view of Made, US Patent Application Publication No. 2002/0056076 (henceforth, "Made '076"). The Examiner's rejection is respectfully traversed.

It has been demonstrated above that independent claims 1, 11, 17 and 20 are allowable in their present form. It follows that claims 5, 8, 9, 12, 13, 18, 19, 21 and 22 that depend therefrom also are allowable.

New Claims

New claims 23-26 add to claims 8, 12, 18 and 21, respectively, the limitation that the attempt to disassemble or convert is initiated at every offset within the suspicious portion(s) of data. Support for these claims is found in the specification as filed at least on page 9 lines 19-23:

> However, in case a vulnerable return address is not detected, and without any advance knowledge regarding any other execution mechanism the attacker is attempting to use, then in order to perform a disassembly and analyze the input stream for malicious code, instruction analyzer 405 needs to perform a disassembly within the suspicious data starting from every possible offset. (emphasis added)

and on page 9 line 30 through page 10 line 1:

> Otherwise, instruction analyzer **405** chooses (step **505**) an offset for example, by enumerating over all possible offsets, and attempts to disassemble (step **507**) the data of the input stream subsequent to the chosen offset. (emphasis added)

In view of the above amendments and remarks it is respectfully submitted that independent claims 1, 11, 16, 17 and 20, and hence dependent claims 2-9, 12-14, 18, 19, 21 and 22 are in condition for allowance. Prompt notice of allowance is respectfully and earnestly solicited.

Respectfully submitted,

Mark M. Friedman
Attorney for Applicant
Registration No. 33,883

Date: September 23, 2009

STRUCTURED COMPUTER ORGANIZATION

FOURTH EDITION

ANDREW S. TANENBAUM

Vrije Universiteit
Amsterdam, The Netherlands

With contributions from
JAMES R. GOODMAN
University of Wisconsin
Madison, WI

PRENTICE-HALL INTERNATIONAL

1.1.3 Evolution of Multilevel Machines

To provide some perspective on multilevel machines, we will briefly examine their historical development, showing how the number and nature of the levels has evolved over the years. Programs written in a computer's true machine language (level 1) can be directly executed by the computer's electronic circuits (level 0), without any intervening interpreters or translators. These electronic circuits, along with the memory and input/output devices, form the computer's **hardware**. Hardware consists of tangible objects—integrated circuits, printed circuit boards, cables, power supplies, memories, and printers—rather than abstract ideas, algorithms, or instructions.

Software, in contrast, consists of **algorithms** (detailed instructions telling how to do something) and their computer representations—namely, programs. Programs can be stored on hard disk, floppy disk, CD-ROM, or other media but the essence of software is the set of instructions that makes up the programs, not the physical media on which they are recorded.

In the very first computers, the boundary between hardware and software was crystal clear. Over time, however, it has blurred considerably, primarily due to the addition, removal, and merging of levels as computers have evolved. Nowadays, it is often hard to tell them apart. In fact, a central theme of this book is

Hardware and software are logically equivalent.

Any operation performed by software can also be built directly into the hardware, preferably after it is sufficiently well understood. As Karen Panetta Lentz put it: "Hardware is just petrified software." Of course, the reverse is also true: any instruction executed by the hardware can also be simulated in software. The decision to put certain functions in hardware and others in software is based on such factors as cost, speed, reliability, and frequency of expected changes. There are few hard and fast rules to the effect that X must go into the hardware and Y must be programmed explicitly. These decisions change with trends in technology and computer usage.

The Invention of Microprogramming

The first digital computers, back in the 1940s, had only two levels: the ISA level, in which all the programming was done, and the digital logic level, which executed these programs. The digital logic level's circuits were complicated, difficult to understand and build, and unreliable.

In 1951, Maurice Wilkes, a researcher at the University of Cambridge, suggested the idea of designing a three-level computer in order to drastically simplify the hardware (Wilkes, 1951). This machine was to have a built-in, unchangeable interpreter (the microprogram), whose function was to execute ISA-level programs by interpretation. Because the hardware would now only have to execute